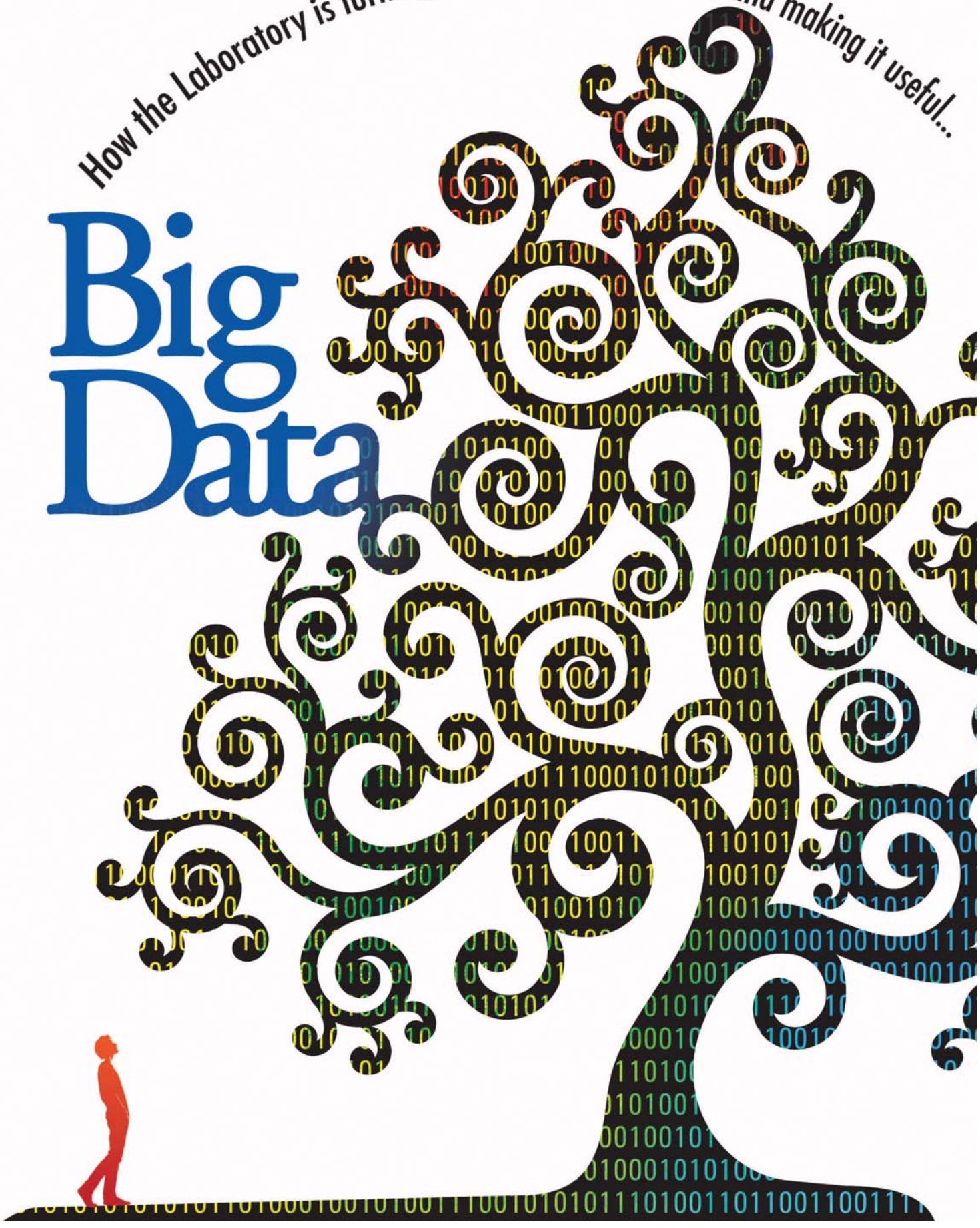
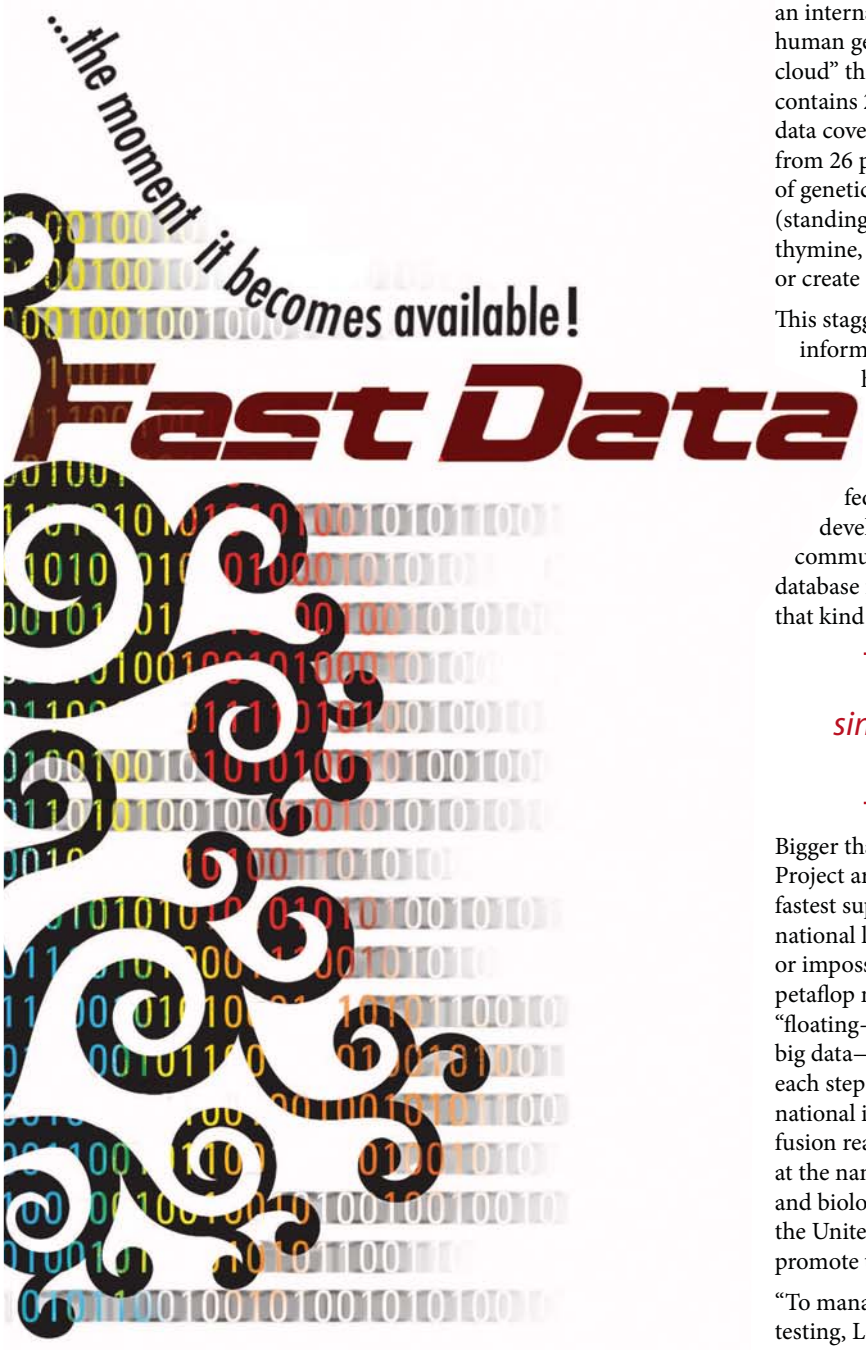
How the Laboratory is turning Big Data into Fast Data and making it useful...

Big Data





Big data is everywhere. Massive sets of digital data are being collected or generated for science, medicine, astronomy and cosmology, national security, cyber-security, situational awareness for our warfighters, social networking, financial markets, and more. And those datasets are big on a scale that boggles the mind.

A good example of big data collected from nature is the recently released database from the 1,000 Genomes Project, an international effort to establish a detailed catalog of human genetic variation. Made publicly available on "the cloud" through Amazon Simple Store Services, the database contains 200 terabytes (200 trillion bytes) of DNA sequence data covering the complete genomes of close to 2,000 humans from 26 populations. If printed as text, these endless strings of genetic code, written in only four letters A, T, C, and G (standing for the four nucleotide bases of DNA: adenine, thymine, cytosine, guanine), would fill 16 million file cabinets or create a paper stack the height of a skyscraper.

This staggering pile of data is a potential gold mine of information for studying such things as differences in human disease resistance and drug metabolism. But can the medical community mine the gold? Does it have the necessary infrastructure and analysis tools for the job? Only recently, because of a $200 million federal big data initiative, were the necessary tools developed and made available to the medical research community for accessing and analyzing the 1,000 Genomes database for insights into human health and disease. It takes that kind of effort to convert big data into *valuable* data.

The national laboratories simulate systems that are otherwise difficult or impossible to test.

Bigger than the dataset collected by the 1,000 Genomes Project are the datasets *generated* by today's largest and fastest supercomputers, which are being used by the national laboratories to simulate systems that are difficult or impossible to test. The laboratories' supercomputers are petaflop machines that achieve more than a quadrillion "floating-point" operations a second (petaflops) and generate big data—hundreds of terabytes of new data—to simulate each step in the dynamic performance of complex systems of national interest. Those systems include the changing climate, fusion reactors and advanced fission reactors, new materials at the nanoscale (one billionth of a meter), complex chemical and biological systems, and nuclear weapons systems, which the United States has not tested since 1992 in order to promote the goals of the Comprehensive Test Ban Treaty.

"To manage the U.S. nuclear weapons stockpile without testing, Los Alamos and Livermore simulate weapons rather than blowing them up, and to achieve the highest-fidelity

simulations possible, we use the largest computers available and generate big data at an ever-increasing scale," explains Gary Grider of the High Performance Computing Division at Los Alamos. "The problem of big data is always about value—about trying to learn something from the data. At that level, we're the same as Google: we want to turn big data into useful information in an affordable and reliable way. And that way must also be scalable—remaining affordable and reliable as datasets continue to grow exponentially."

But are the national labs getting the most out of this big weapons simulation data from the latest supercomputers? And are they ready with the data management and analysis tools to handle the much larger datasets that will be produced by the next generation of machines?

To achieve the highest fidelity simulations possible, we use the largest computers available and generate big data at an ever increasing scale.

The current answer is a big NO! Unlike the 1,000 Genomes Project big data initiative, the initiative for big weapons data is nowhere near complete, but it has been going on quietly behind the scenes at Los Alamos for almost a decade.

The Big Data Bottleneck

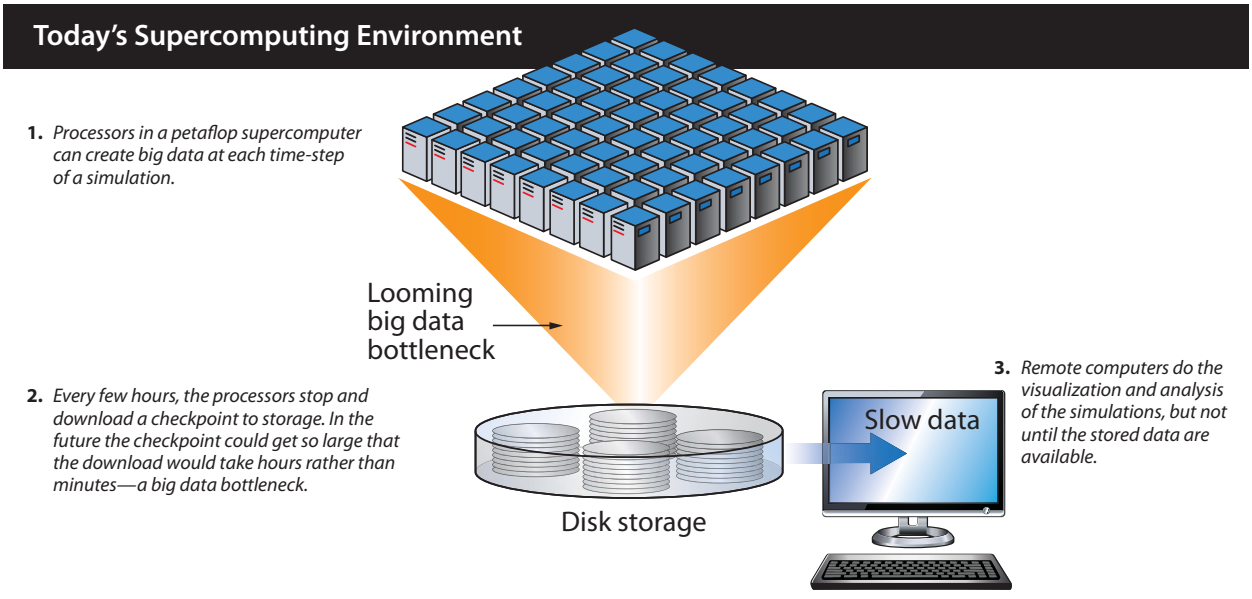
For the past 20 years, supercomputers have generated ever-more simulation data at ever-faster speeds, but those data are not useful until they are selected and moved to permanent storage, organized into files, and then accessed by auxiliary computers that analyze the data and create visualizations of the simulated systems. All those data-handling steps are being challenged by big simulation data, but the biggest challenge is the growing mismatch between the rate at which supercomputers generate data and the rate at which those data can be transferred from the supercomputer to magnetic disk storage, the best permanent storage around. Like cars trying to exit a five-lane highway by way of a narrow ramp, big simulation data of the future will hit a big bottleneck in the transfer path between the supercomputer and storage (see figure below).

Without a solution, computing in 2020 will see crippling data traffic jams in which exaflop supercomputers are idle half the time.

To be specific, the supercomputer world is racing to increase calculation speed a 1,000-fold by 2020—from petaflops to exaflops (a quintillion operations a second)—whereas data-transfer rates to disk storage are expected to increase only 30-fold by that year. Without a solution to this growing mismatch, computing in 2020 will see crippling data traffic jams in which exaflop supercomputers are idle half the time, bloated with data stuck at the bottlenecks separating data generation from data storage and analysis.

Computing at the exascale has often been viewed as a holy grail. For the national security labs, that is because exascale is the scale at which high-fidelity, 3D weapon simulations become practical (see "Will It Work?" in this issue). But the closer supercomputing speeds get to the exascale, the larger the specter of big data becomes. To prepare for the next-generation computers and ensure that they live up to their promise, Grider and colleagues are working closely with

Today's Supercomputing Environment



industry and coming up with affordable, scalable solutions. These will not only relieve the big data bottlenecks to disk storage but presage a more effective approach for managing big data simulations at the exascale and beyond.

How Big Simulations Get Done

To better understand these big data solutions, you have to know how today's high-performance computers work. These machines are massively parallel: they can contain more than a million processors, and all million-plus of them work in tandem on tiny bits of the same simulation.

Suppose the simulation is needed because a killer asteroid, one the size of the Rose Bowl, is on a collision course with Earth, and the government wants to know if a nuclear detonation can destroy it. This scenario cannot be tested in a laboratory. But it could be simulated on a supercomputer to help predict whether a nuclear detonation would succeed (see "Killing Killer Asteroids" in this issue).

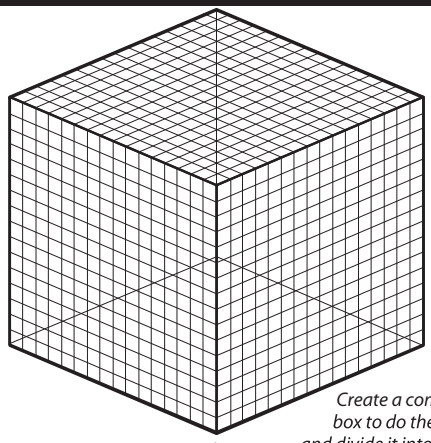
The simulation might run from weeks to months. And the work is never smooth going.

To do the simulation, a model of an asteroid is placed in a computational box (a way to specify the 3D coordinates of every point in the asteroid model). In this case, the supercomputer is to simulate the entire event, that is, compute all the heating, vaporizing, fracturing, and accelerating, along with the final trajectories of the asteroid fragments, that result from the blast wave from a nuclear detonation hitting the asteroid.

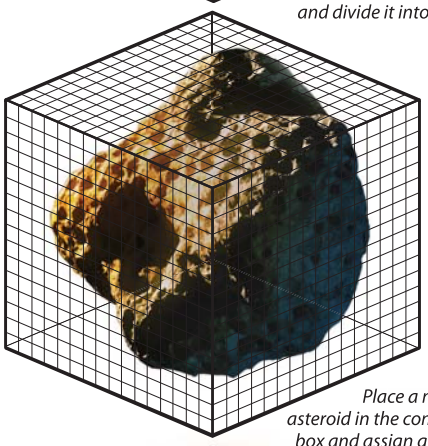
To simulate that event on a modern supercomputer, the computational box is divided into a 100 million smaller cubes of equal size, just as a Rubik's cube is divided into smaller cubes. Groups of the small cubes are assigned to different processors, and each processor solves the physics equations describing what the blast wave does to the material in its set of cubes. The event's duration is divided into discrete time steps (say, several microseconds long), and together, the processors simulate the event one time step at a time. When a processor computes that fragments of rock and vaporized rock in one of its assigned cubes are crossing into a neighboring cube, the processor must pass its latest data about their position, density, temperature, velocity, and so on to the processor for the neighboring cube.

Even though all the processors are sharing the computational load, each processor must solve complicated sets of physics equations for each of the hundreds of thousands of time steps, so the simulation might run for weeks to months to reach completion. And the work is never smooth going. A petaflop computer has millions of parts connected by miles of cable, and a processor fails on average every 10 to 30 hours,

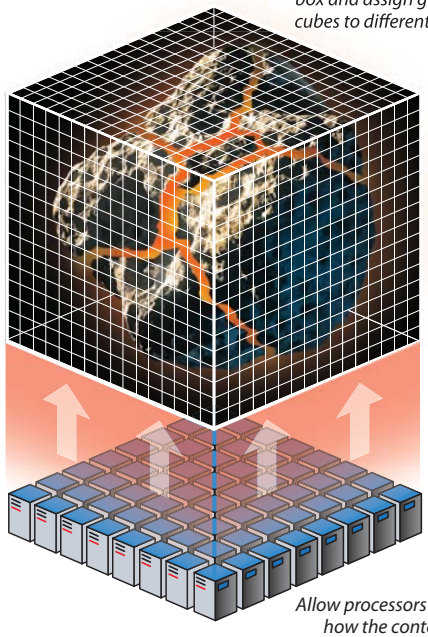
How a Simulation Is Done



Create a computational box to do the simulation and divide it into 100 million cubes.



Place a model of the asteroid in the computational box and assign groups of the cubes to different processors.



Allow processors to compute how the contents of each cube evolve.

corrupting some of the data needed for the next time step. And because what happens in one cube depends on what comes in from and goes out to neighboring cubes, all the processors must work cooperatively. The failure of one processor has a domino effect: when one stops, all the rest must stop. Does that mean the simulation must return to "start" each time a failure occurs? That would be like writing a document and never using the "Save" command—a very dangerous strategy.

Instead, a supercomputer has to play defense. Every 4 hours, it stops and creates a checkpoint, the analog of pressing "Save" or taking a snapshot of the simulation. All the processors stop at the same simulation time step; update the data describing the temperature, pressure, position, velocity, and so on of materials in their cubes; and send the data to the storage system, which is outside the main computer. Thus, whenever one or two processors fail and the computer crashes, the computer automatically stops, retrieves the data from the nearest checkpoint, and resumes computing at that point. These reference checkpoints not only provide a backup but also record the calculation's progress.

A supercomputer has to play defense. Every 4 hours, it stops and creates a checkpoint, the analog of pressing "Save."

Storing checkpoints sounds simple, but a petaflop supercomputer must save as many as 50 to 100 terabytes of data for each checkpoint, so this kind of "Save" can be very costly in time. Grider explains, "The disk drive in your computer at home might have 1 terabyte of storage capacity, and it would take you about 11 hours of writing to fill that up. We need to transfer all 50 to 100 terabytes in about 5 minutes because while we're writing to memory, we're not getting any science done. So we need 10,000 disk drives hooked together to transfer the checkpoint data to all the storage disks in parallel and get the job done in minutes." On today's petaflop machines, the job does get done, but barely.

Years ago, Los Alamos anticipated that its next big development after Roadrunner, the first petaflop machine, would be Trinity, which, at a speed of 40 to 100 petaflops, would need to store 2 or 3 petabytes of data at each checkpoint. That would require buying 30,000 disk drives at a cost of $30 million, or 20 percent of the machine's cost, and they would be difficult to maintain. An exascale machine would need about 100,000 disk drives, costing 40 to 50 percent of the machine's cost; that would be unaffordable. Without those disk drives, it would take an hour or two to dump the data at each checkpoint, so a major fraction of the computing time would be lost to defensive storage. Neither option was acceptable and both would get worse over time. "Our only course," says Grider, "was to initiate research and development with government, academia, and industry and find an affordable, scalable way around the big data bottleneck."

Burst Buffers—From Big Data to Fast Data

The bottleneck problem that Los Alamos is solving with industry is two-fold: *decreasing* how long processors remain idle when transferring checkpoint data to storage and *increasing* how quickly checkpoint data is fed back to the processors when they fail.

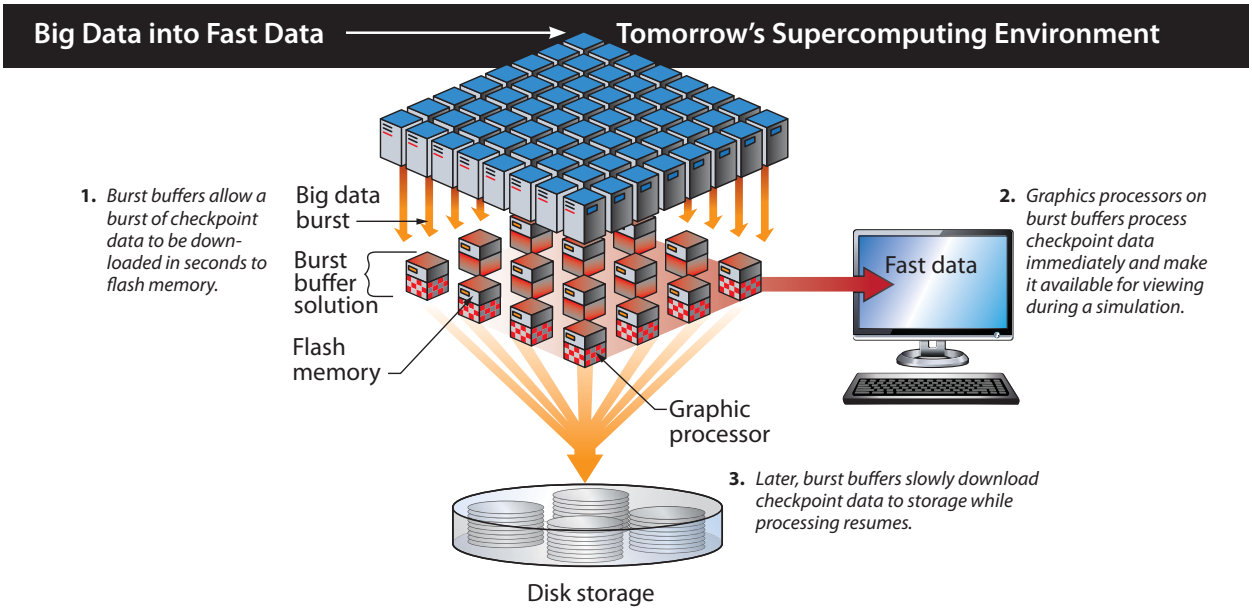
The solution that is in the works capitalizes on flash memories—solid-state storage devices that can write (store) data about 10,000 times faster than disk drives can. If flash memories are placed between the processors and the disk storage, they can "buffer" the mismatch between the burst of checkpoint data needing to be downloaded very quickly and the disk drives, which write data slowly. Grider coined the name "burst buffer" to describe the device that will hold this rapid-writing flash memory and have the right connections to both the supercomputer and the disk storage.

Grider explains, "The concept of the burst buffer is to have the burst of data written onto flash very quickly and then have it written from flash to disk slowly. That way you don't need so many disk drives, and you use the storage disks for what they're good at, namely capacity storage—storing large quantities of data securely."

Imagine racks of processors that are doing the simulation and beneath them the permanent storage system. Each of the million processors is connected to one of many thousands of burst buffers that together act as a staging area to hold checkpoint data before they are sent to permanent storage (see figure, opposite page). An entire checkpoint in the form of a huge petabyte data stream—a burst of data—is downloaded from all the processors in parallel and is absorbed in seconds by the flash memories in the buffers, the processors then resume the simulation. Later, the checkpoint is drained from the burst buffers to disk storage, but at the much slower rate that the disk drives can handle. That means that the processors are stopped so briefly for the downloading to flash that they run almost continuously, with data being written from flash to disk in the background while the processors keep doing science.

We'll be able to watch the simulation as it's happening and intervene if we see something that needs changing. This is truly big data becoming fast data.

Further, if one adds two flash memory units to each burst buffer, one of those units could hold onto the most recent checkpoint data for hours, and download it to disk storage *only after* the second flash unit had received data for a new checkpoint from the processors. Because data downloading would toggle between units, a complete checkpoint would always be available in the burst buffers, ready to be fed back



to the processors if a failure required the simulation to be restarted. Flash would virtually eliminate delays caused by both a processor failure and a slow-moving "Save."

Once burst buffers have enough flash memory units to temporarily store checkpoint data, it becomes possible to add graphics processors to each burst buffer. Then, instead of waiting until the end of a run for a visualization of the completed simulation, the current situation, the checkpoint data could be processed into a visualization while the simulation was in progress!

Los Alamos knew back in 2006 it needed to innovate, and it came up with a winner.

"That means we'll be able to watch the simulation as it's happening and intervene in the middle of a run if we see something that needs changing," says Grider. This is truly big data becoming fast data—useful at the moment it becomes available.

"This is the beginning of a big story," continues Grider. "Adding graphics processors to the burst buffer is an example of what's called 'process-in-memory'—processing data where it's most appropriate. Today we move the data to the processors, do the math (addition, multiplication, whatever it is), and then write the results back out to memory [storage]. But the time it takes to move the data is wasted because it's time in which no computing is going on. It may take less time to ship the process to where the data is, and that's what we'd be doing by shipping analysis and visualization to a processor in the burst buffer. So the big story is that processing in the future could go on wherever there's data—in memory, in flash, near disk, near tape. That way some of the processing for a big simulation can take place off the main computer." That is how big data will become fast data.

For weapons simulations, the burst buffer idea is great because it not only allows the downloading or uploading of big data in a few minutes, but it also enables big data to be processed during the simulation, making it useful data.

Race to the Exascale

The Laboratory was driven to develop the burst buffer so it can to do high-resolution 3D simulations of nuclear weapon detonations at the exascale by 2020, but it also needs it because of the constraints of performing exascale simulations affordably and within practical time limits. Los Alamos knew back in 2006 that big data at the exascale would lead to big data bottlenecks and make the old way of doing supercomputing unaffordable. It knew it needed to innovate, and it came up with a winner.

"The burst buffer with its flash memory is the only way we'll be able to build a cost-effective exaflop machine in the 2020 time frame," explains Grider, "and we'll be trying it out on Trinity in the 2015–2016 time frame. Then, when we really need it, we'll have it working. And even as early as Trinity, we'll be testing burst buffers with processors that can analyze and distill the data while the simulation is running." And that's not all. According to Grider, Trinity will be a testbed not only for the burst buffer, but for debugging some of the software Los Alamos will need to keep an exaflop machine running smoothly.

Los Alamos is doing serious prepping for the exascale.

~Necia Grant Cooper